Watermarking technology used in CBIR

Henrik Skibbe

Seminar CBIR

Watermarking technology used in CBIR

- **Introduction**
- **Digital watermarks**
- **Embedding an invisible & robust watermark**
- **Watermarks & CBIR**
- **Used features**
- **Test results**
- **Conclusion**

Introduction

Digital media is largeley distributed

INTERNET
CD
DVD

Everybody can make lossless and unlimited copies of digital contents.

Introduction

Finding methods for

copy protection
copyright protection
authentification
searching information

Introduction (protection)

Conventional cryptographic systems permit only valid keyholders access to the encrypted data

Problem:

While our data is encrypted nobody can access them, once they are decrypted there is no way to avoid reproduction.

Introduction

Solutions :

digital watermarks

visual information retrieval

Digital watermarks

Digital watermarks

There are two kinds of watermarks:

visible

invisible

Digital watermarks

A visible watermark contains a visible message or a logo. A key is necassery to remove it from the marked image.

Useful for :

demonstration, indicating the ownership

Digital watermarks

More interesting are the invisible watermarks.
There are two classes of watermarks:

fragile

robust

Digital watermarks

Fragile watermarks

fragile to most modifications

useful for content authentication & integry attestation

Digital watermarks

Robust watermarks

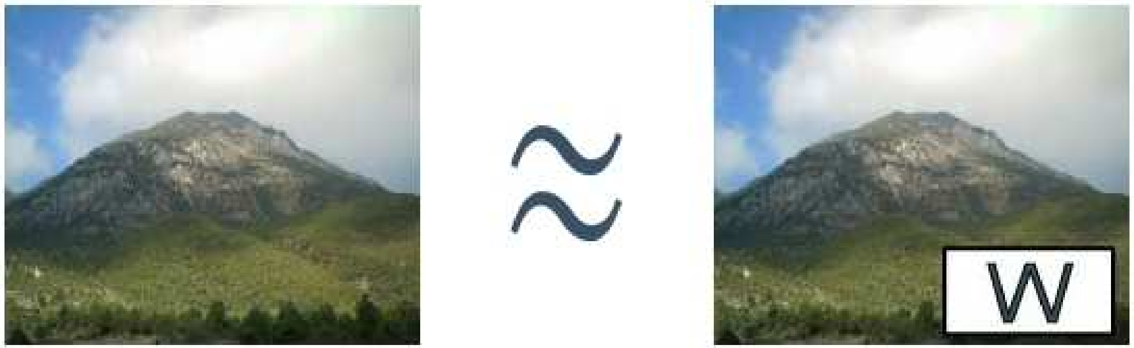
robust to nearly any kind of image processing operations, like

cropping, blurring, compressing

used for copyright protection & ownership verification

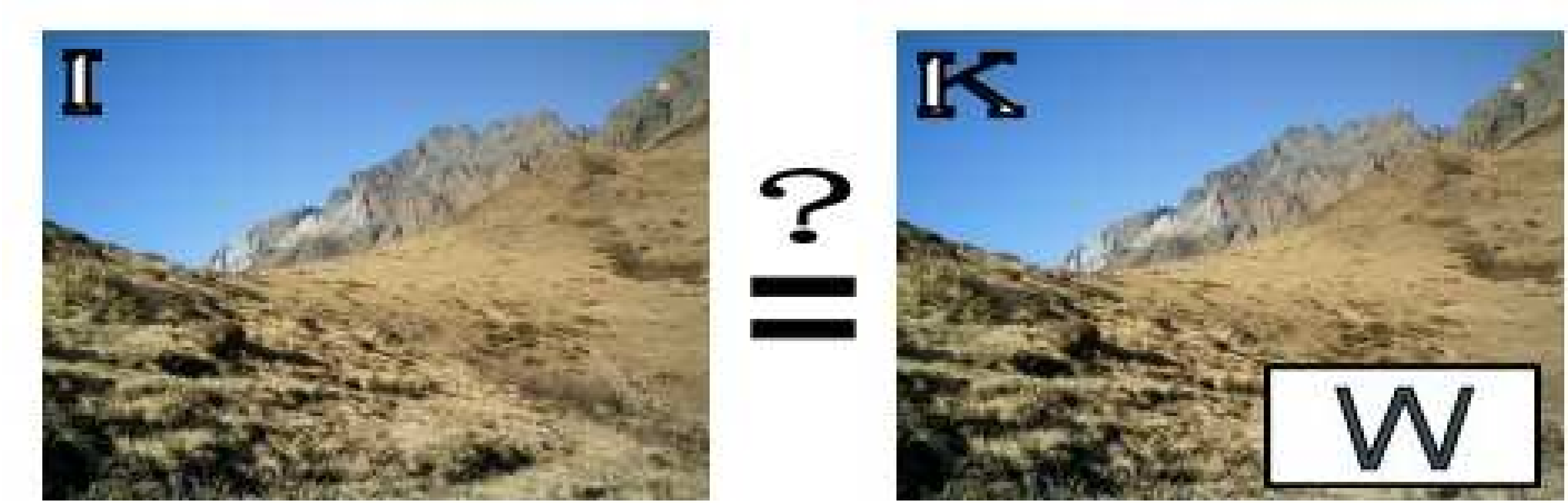
embedding an invisible & robust watermark

Our goal:
similar as possible



& robust

Mean Squared Error



$$MSE = \frac{1}{mn} \sum_{i}^{m} \sum_{j}^{n} ||I(i,j) - K(i,j)||^2$$

Peak signal-to-noise ratio

$$PSNR = 10 \cdot \log\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot \log\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

MAX is the maximum pixel value of the image. f.ex. 255

The color space YUV

Converting the image from RGB to YUV color space. We use the Y (brightness) channel to store our watermark in.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.148 & -0.289 & 0.437 \\ 0.615 & -0.515 & -0.100 \end{pmatrix} * \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

- Robust against modifing the image's colors
- Robust against JPEG compression

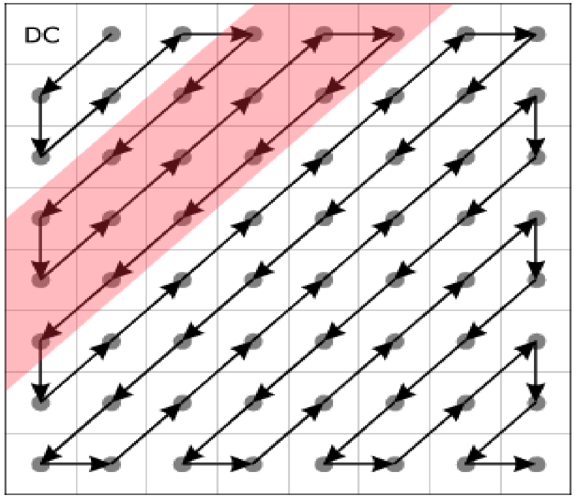
Where to store hidden informations ?
Using the DCT domain !

- Changing coefficients is less „visible“ & much more robust than directly changing a pixel’s value
- Part of the JPEG compression method
- There are fast algorithms for DCT & IDCT

$$F_{x,y} = \frac{C(x) \cdot C(y)}{4} \cdot \sum_{i=0}^{7} \sum_{j=0}^{7} f_{i,j} \cos\left(\frac{(2i+1) \cdot x \cdot \pi}{16}\right) \cdot \cos\left(\frac{(2j+1) \cdot y \cdot \pi}{16}\right)$$

$$C(n) = \begin{cases} \frac{1}{\sqrt{2}}, & n = 0 \\ 1, & n \neq 0 \end{cases}$$

Dct coefficients



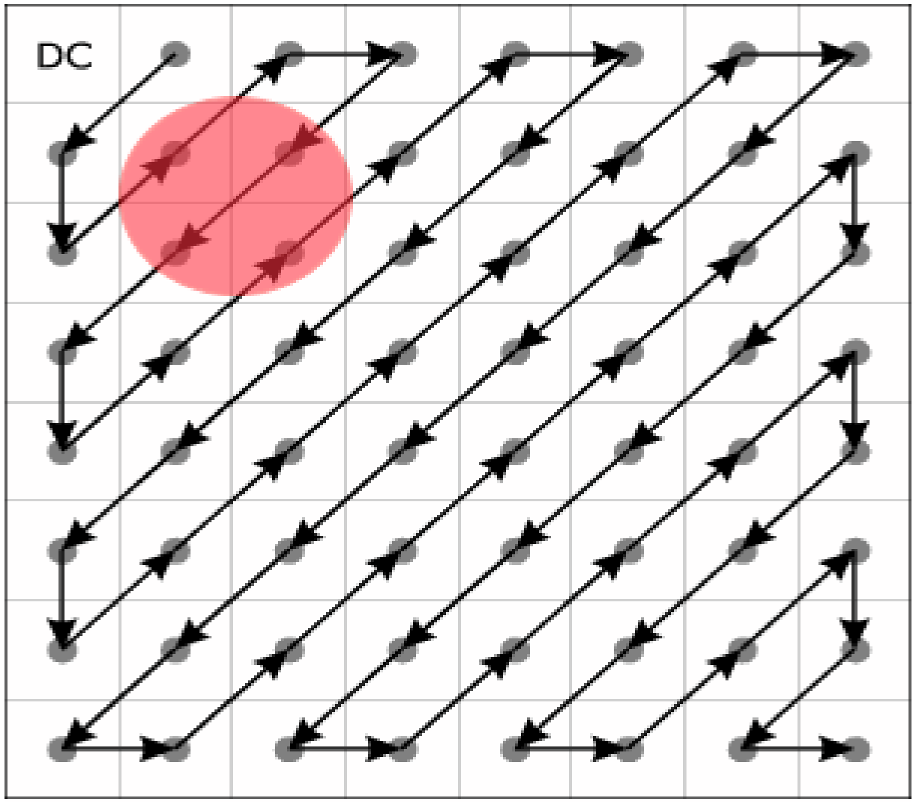
→We will use 8x8 blocks for embedding bits of our watermark

→changing lower frequencies are more robust against JPEG compression

→But also have the most influence to the image's quality !!

→We will not change the DC coefficient, so the image is robust against changing its brightness

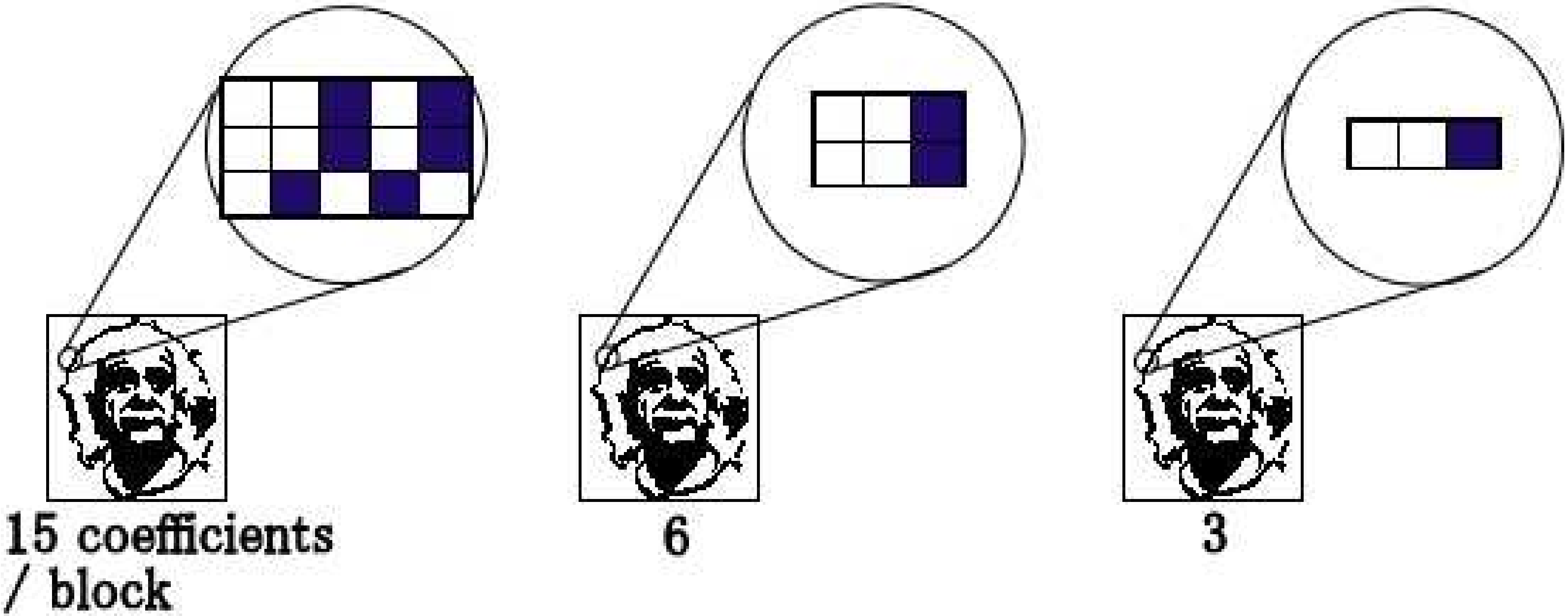
Dither modulation



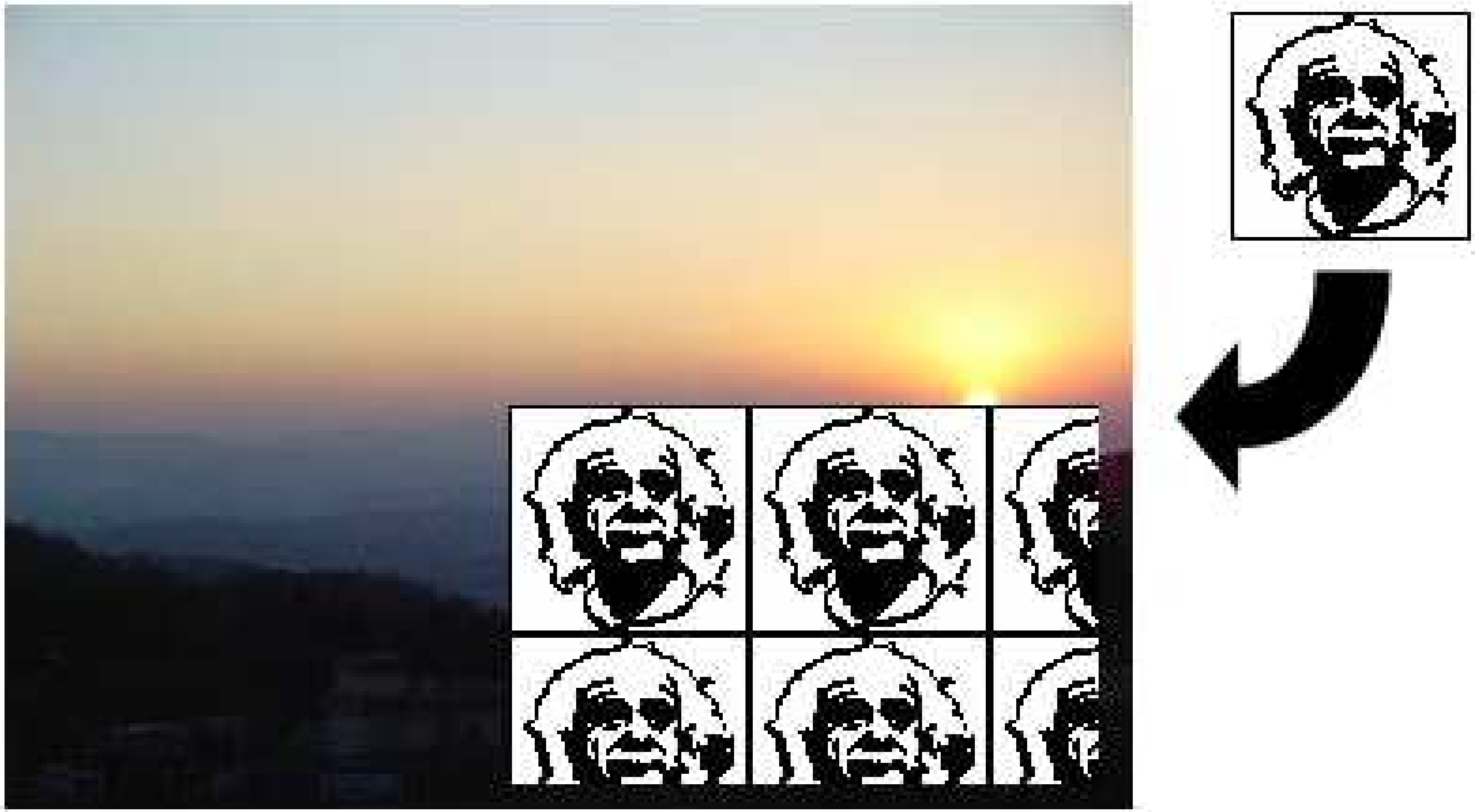
4 DCT coefficients, so we can encode 4 hidden bits into each 8x8 block

Using one DCT coefficient to encode one bit.

Bit alignment

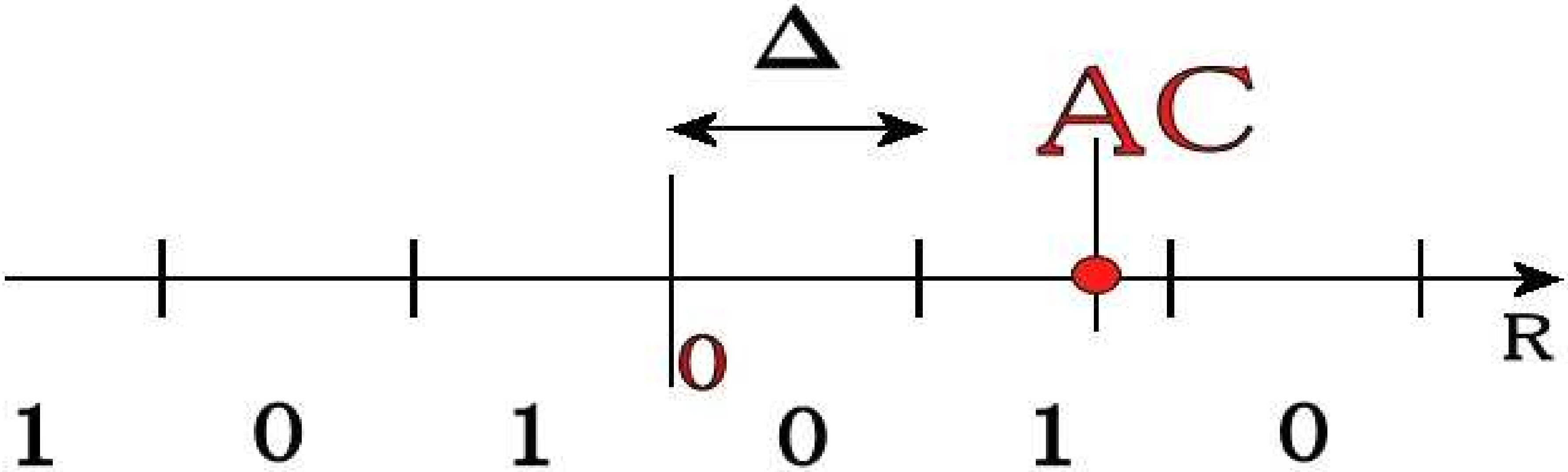


Bit alignment



- robust against cropping
- redundant bits

Dither modulation



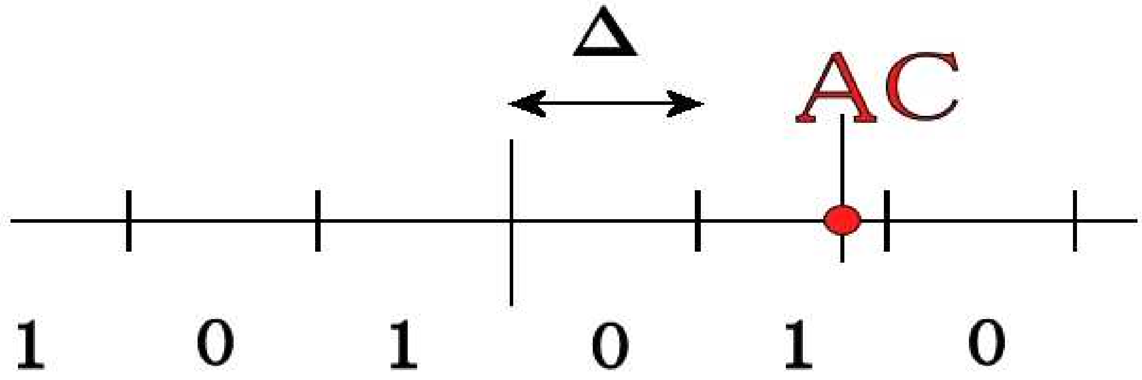
* even intervals are representing a 0
* odd intervals a 1

an interval is even, if

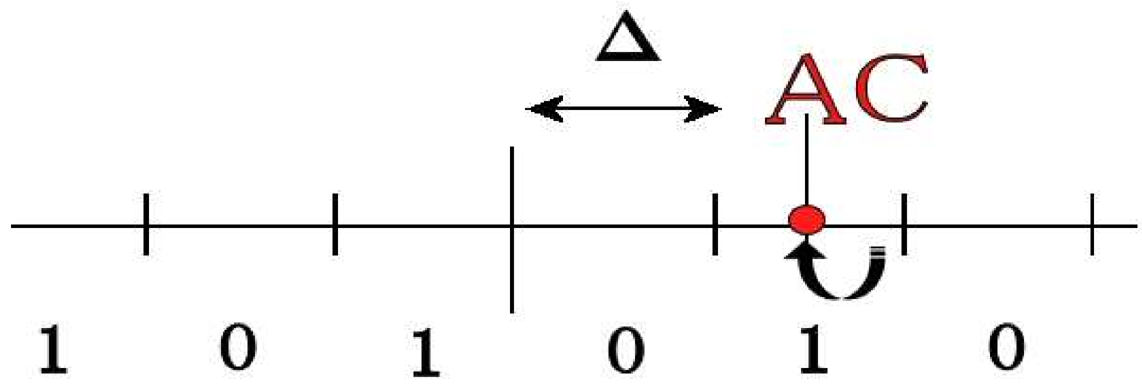
$$\lfloor \frac{x}{\Delta} \rfloor = even, \forall (x \in I \wedge x \geq 0)$$

Dither modulation

A DCT coefficient should represent a "1" and is already in an interval which represents a "1".

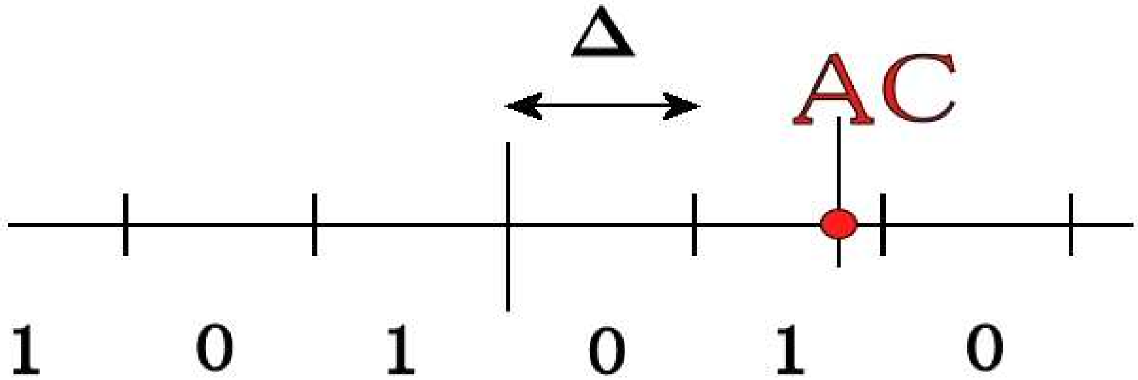


To make its information more robust, we move it into the middle of the interval

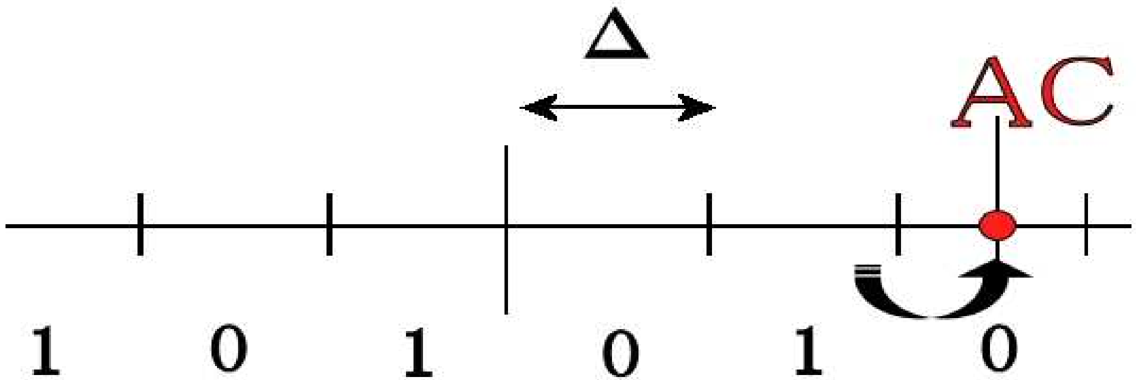


Dither modulation

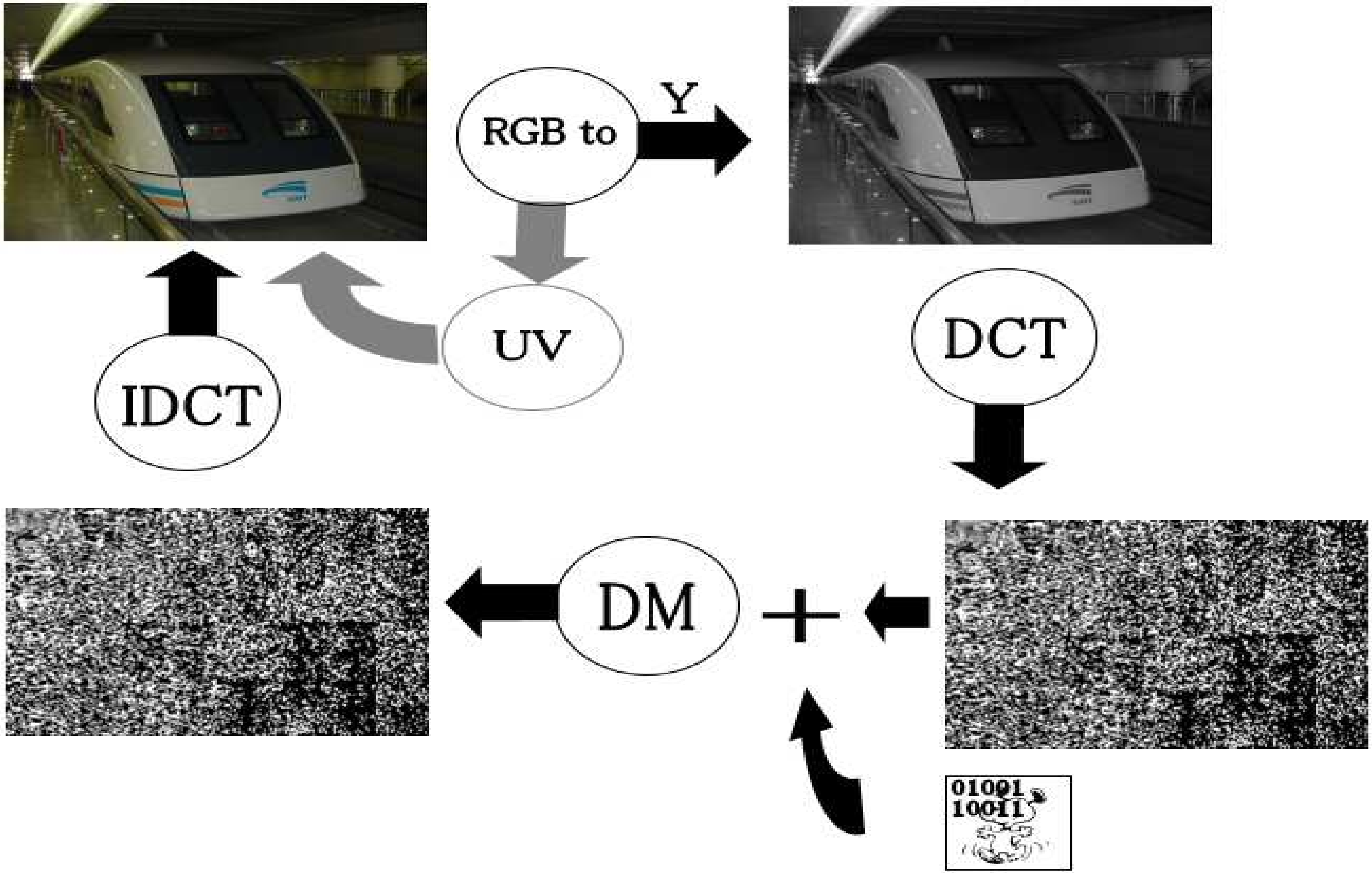
A DCT coefficient should represent a ″0″ but is in an interval which represents a ″1″.



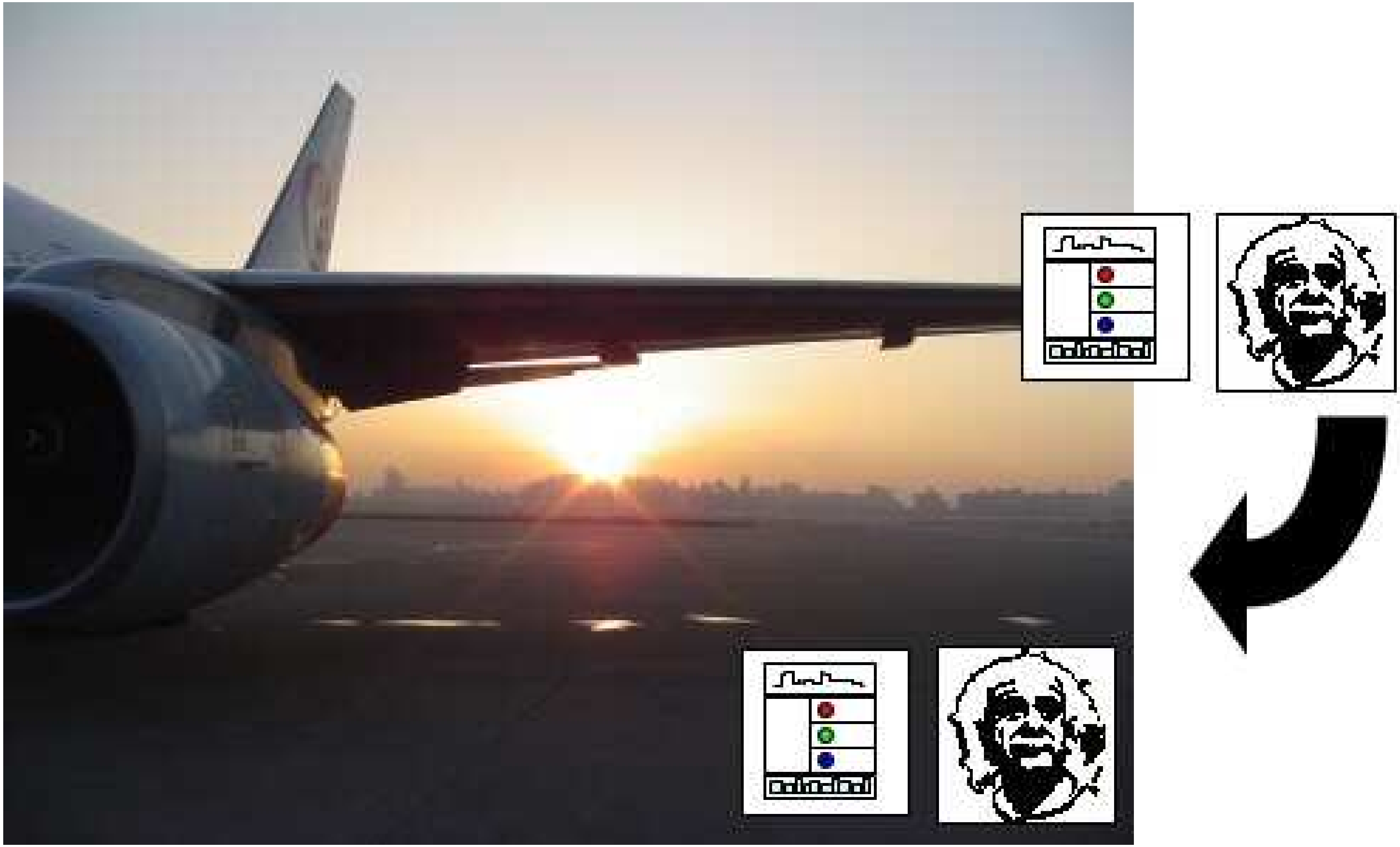
We move it to the nearest interval which represents a ″0″



Watermarking process



Watermarks & CBIR



Watermark & CBIR

Idea :

embedding image related information into the image using the watermarking technique

Watermark & CBIR

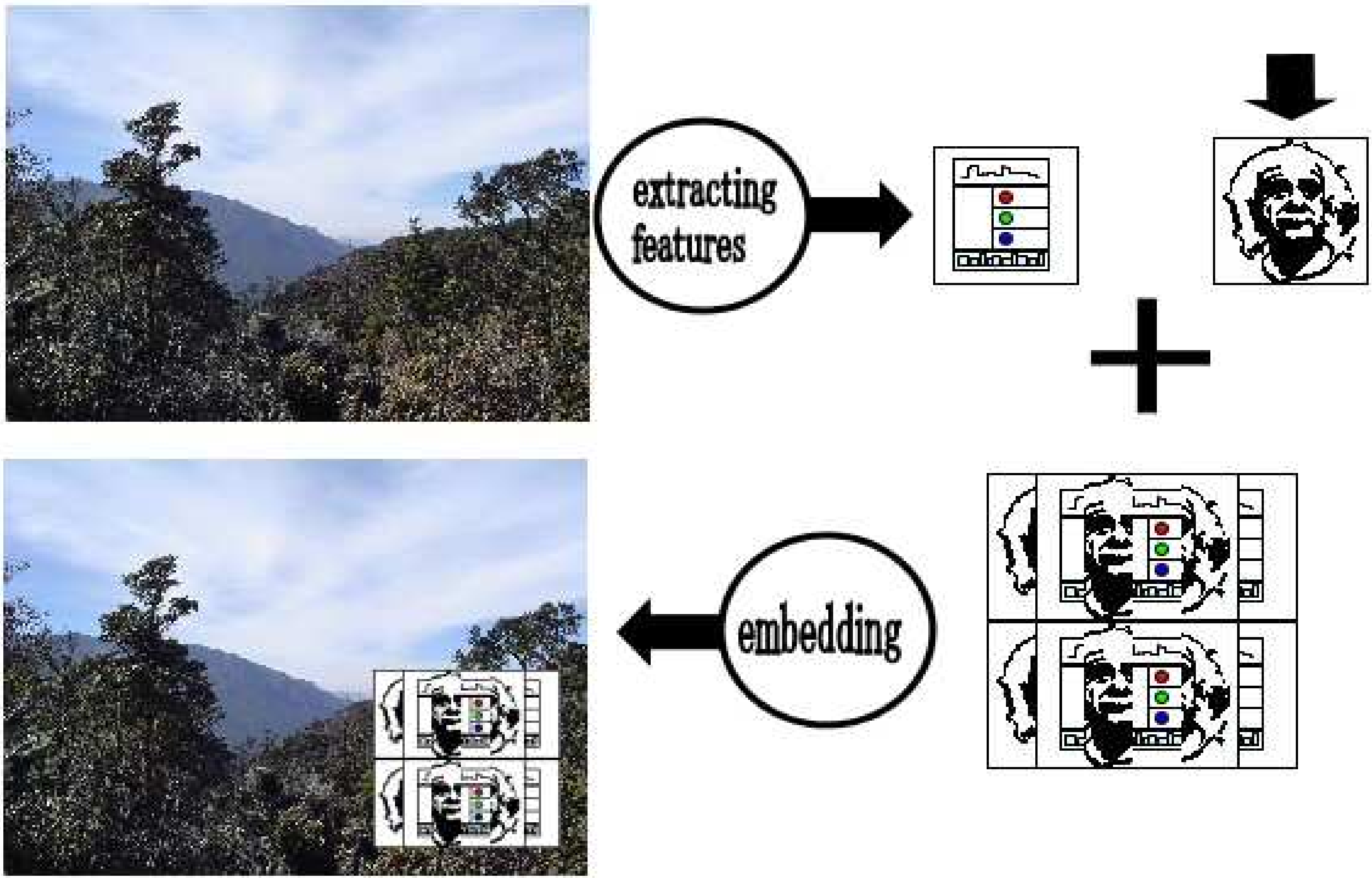
Concrete:

extracting features from the image and save them into a feature vector.

creating a watermark, which bits are identical to it.

embed the watermark

Watermarking & CBIR



Used features

Haar Integral

$$A[f](\mathbf{X}) = \frac{1}{2\pi NM} \int_{t_0=0}^{N} \int_{t_1=0}^{M} \int_{\varphi=0}^{2\pi} f(g\mathbf{X})\, d\varphi\, dt_1\, dt_0$$

statistic moments

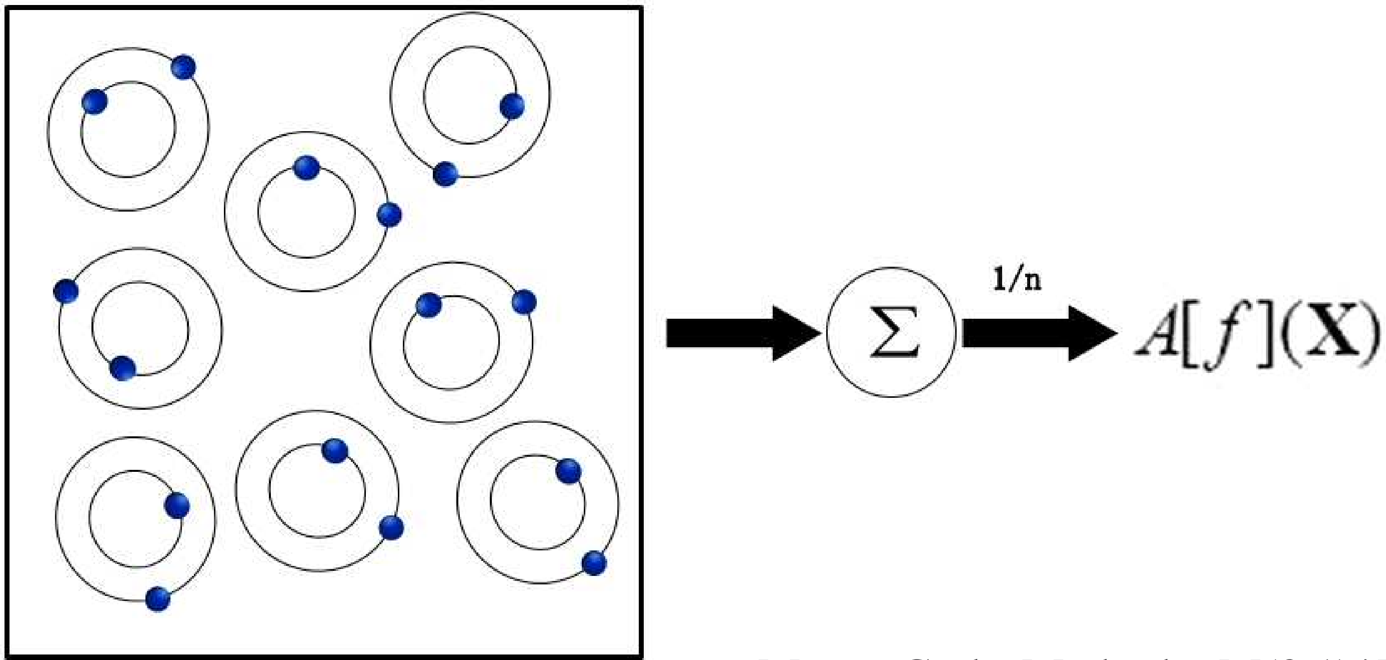
$$M_k(r) = E\left(\left(X - r\right)^k\right)$$

Hu moments

$$m_{pq} = \sum_{i=1}^{N_x} \sum_{y=1}^{N_y} x^p y^p I(x, y)$$

Haar integral

$$A[f](\mathbf{X}) = \frac{1}{2\pi NM} \int_{t_0=0}^{N} \int_{t_1=0}^{M} \int_{\varphi=0}^{2\pi} f(g\mathbf{X}) d\varphi dt_1 dt_0$$



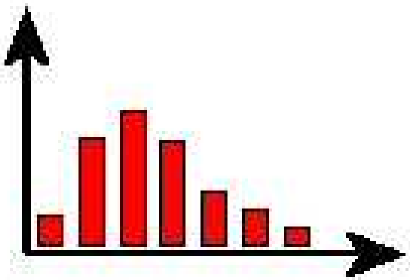
Monte-Carlo-Method, M(0,1)*M(2,0)

Statistic moments

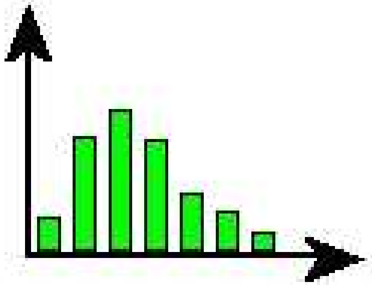
$$M_2(\mu) = E((X-\mu)^2)$$

$$s = \frac{M_3(\mu)}{\sigma^3}$$

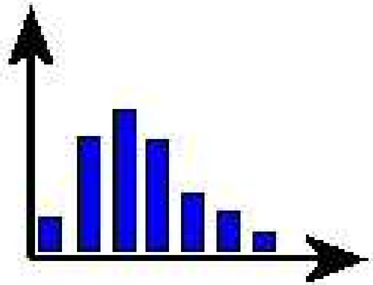
$$k = \frac{M_4(\mu)}{\sigma^4} - 3$$



$[E(X), VAR(X), s, k]^T$



$[E(X), VAR(X), s, k]^T$



$[E(X), VAR(X), s, k]^T$

Hu moments

$$m_{pq}=\sum_{i=1}^{N_x}\sum_{y=1}^{N_y} x^p y^p I(x,y)$$

$$\mu_{pq}=\sum_{i=1}^{N_x}\sum_{y=1}^{N_y} (x-\bar{x})^p (y-\bar{y})^q I(x,y)$$

$$\eta_{pq}=\frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \gamma=\frac{p+q+2}{2}$$

$$\phi_1 = \eta_{20} + \eta_{02}$$

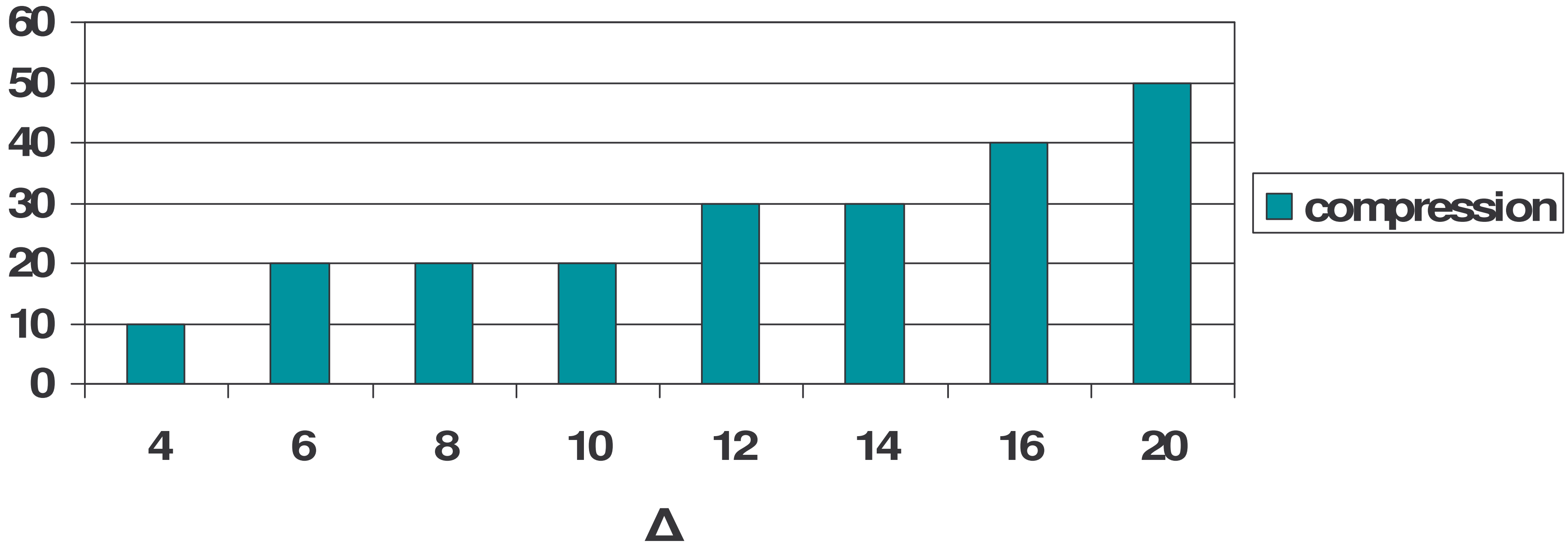
$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

Test results

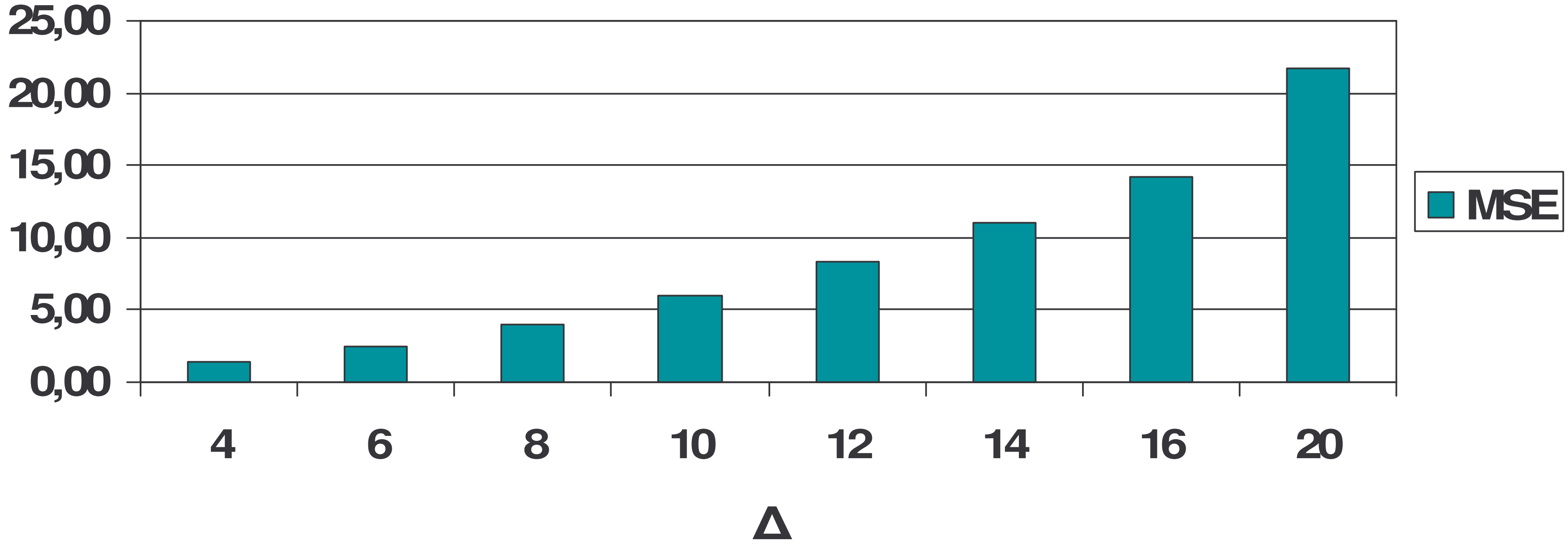
Interval size / compression



Retrieved watermark > 85 % identical to the original information

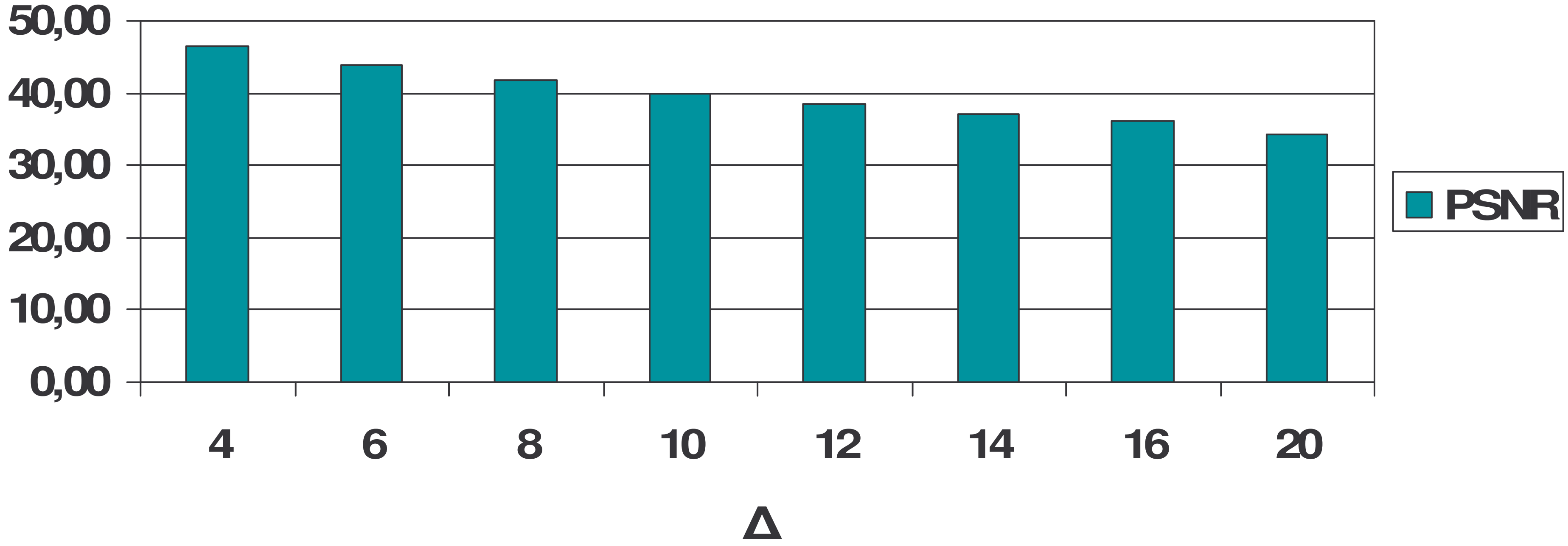
Test results

Interval size / MSE



Test results

Interval size / PSNR



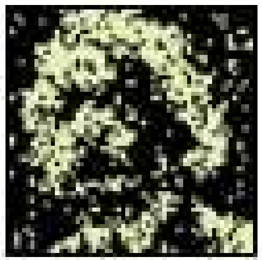
Robustness against attacks

Test image : 384 x 256, marked with a 48x48 bit watermark

Cropping

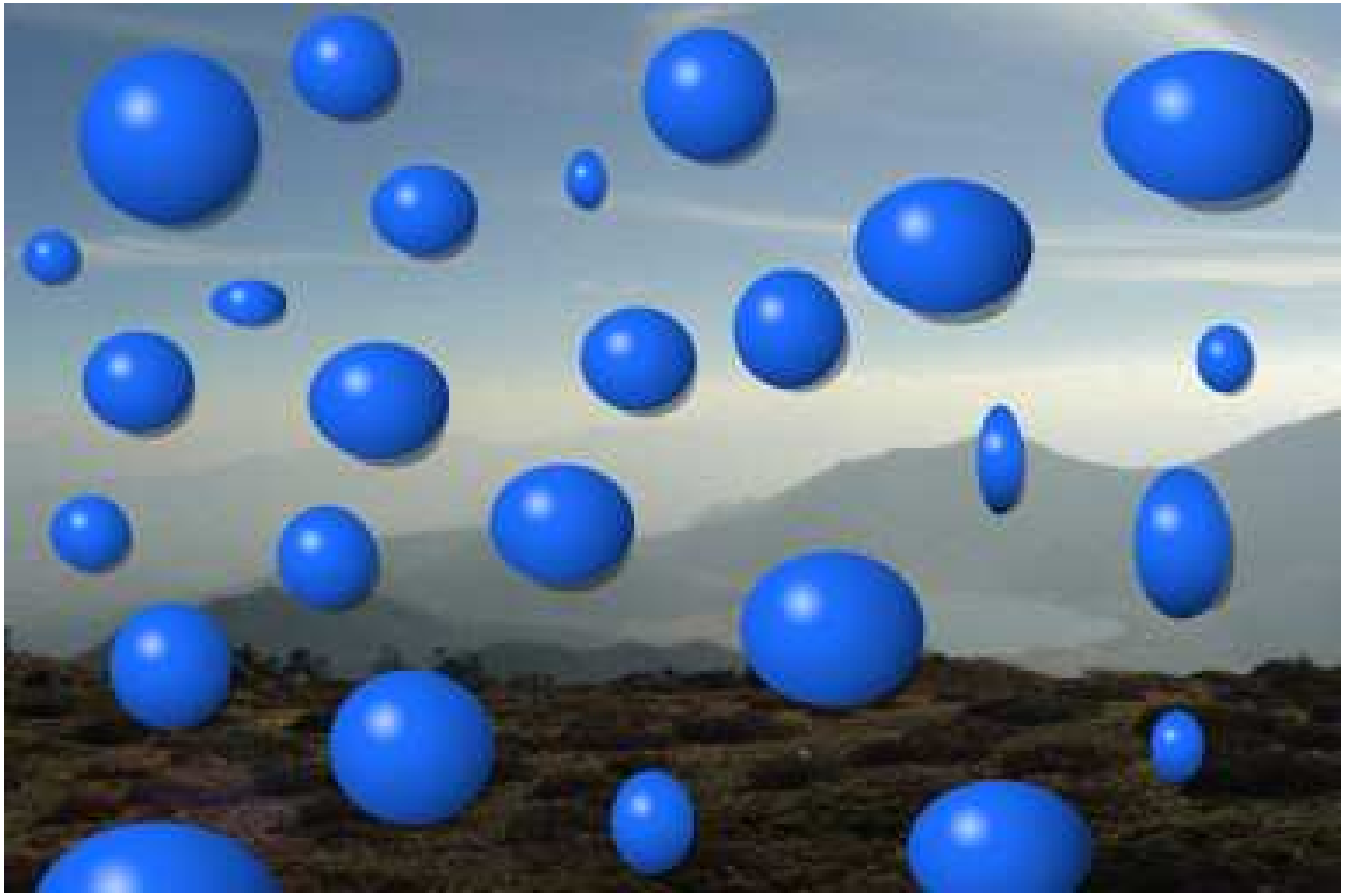
100%

Cropping 2



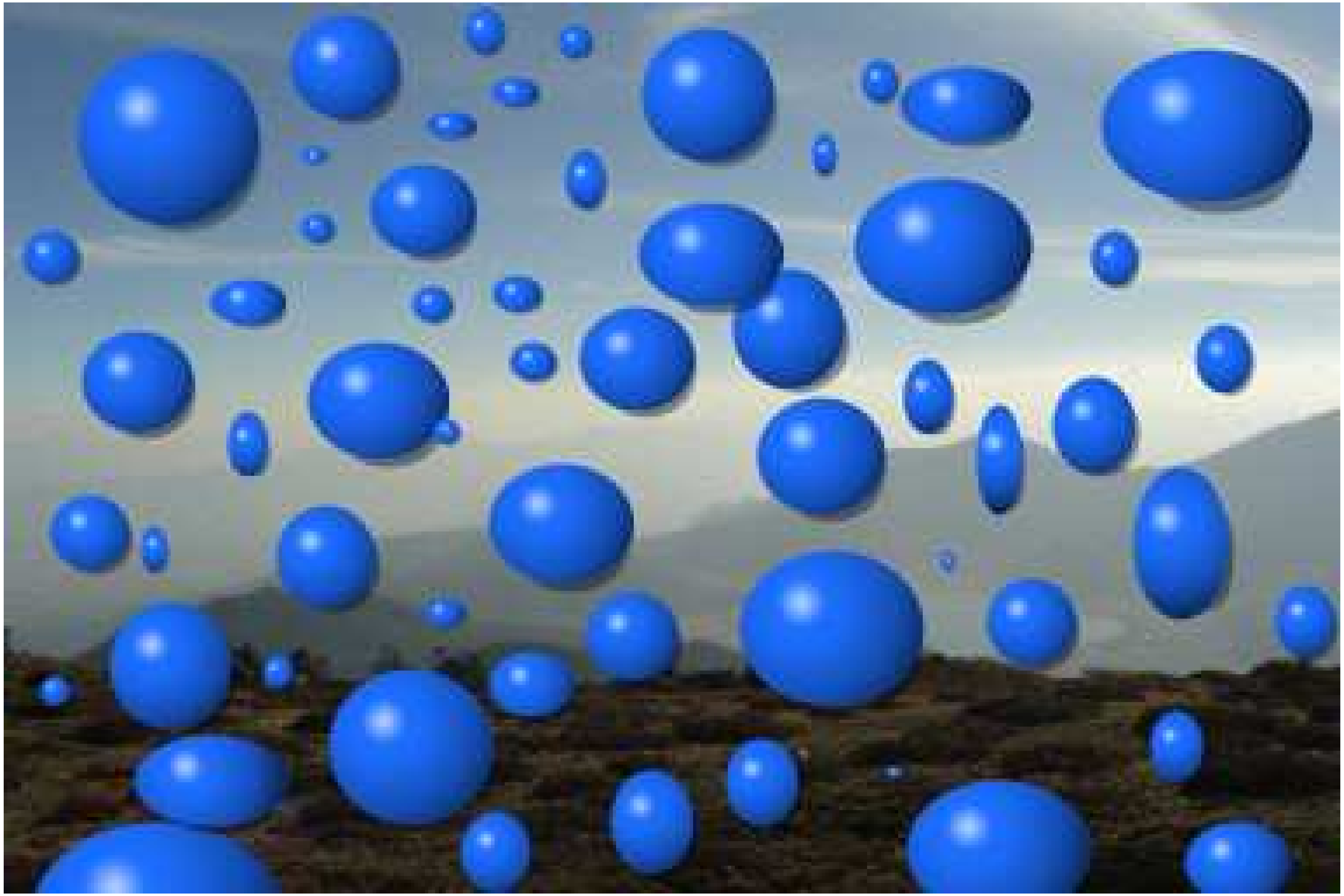
83,8%

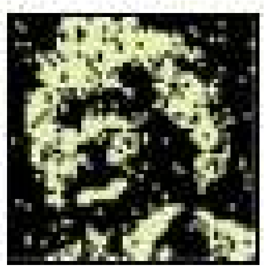
„Blobs“



97,6%

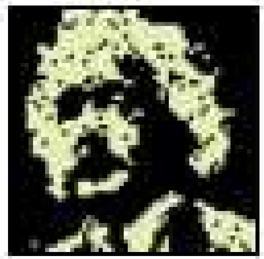
More „Blobs“ ☺





89,7%

Brightness – 25%



96,1%

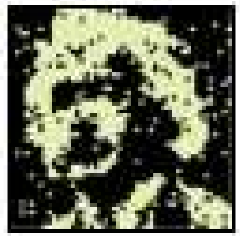
Brightness + 25%

100%

BW

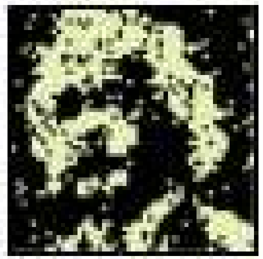
100%

Color



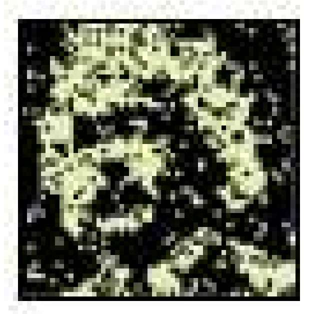
93,4%

Color 2



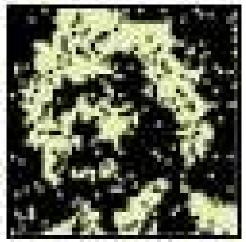
91,6%

Gaussian blur, radius 1,0



85,2%

Median filter, aperture 3



89%

negative

0%

Conclusion

- Invisible watermark
- Robust against cropping, blurring ,
  [translation, scaling] , …..
- Easy to implement & fast
- No extra space for feature vector needed

- Not robust against rotation

☺